BÖLÜM 4

4. VERİ TABANI TASARIMI ve NORMALİZASYONU

4.1. Veri Tabanı Tasarımı

Iyi bir veritabanı tasarımı yapabilmek için yetenek, bilgi ve tecrübe çok önemlidir. Öncelikle, ilişkisel veritabanının tanımını ve bununla ilgili 5 Normalizasyon kuralını çok iyi bilmek gerekir. 5N, tasarım aşamasında yol göstermek yerine hangi şartlara uygun tasarım yapılması gerektiğini anlatır. Bazen, bu kurallardan vazgeçmek durumunda olunabilir ancak, veritabanında saklanacak verilerin hacmi arttıkça yani veri tabanı büyüdükçe bu kuralların daha sıkı uygulanmasın gerekir.

Bir veri tabanı ile proje yapılırken işin en önemli aşaması veri tabanının tasarlanmasıdır. Başlangıçta yanlış tasarlanan bir veri tabanı ile yapılan projede sonradan yapılacak düzenlemelerle geri dönüş yapılamaz. O nedenle Veri tabanı tasarımı yapılırken aşağıdaki maddelere uyularak yapılması gerekir.

1. Nesneler Tanımlanır: Nesne, çeşitli özellikleri bulunan bir varlıktır. Herhangi bir proje de öncelikle nesneler tanımlanır. Birkaç proje için nesnelere örnek verilecek olunursa,

Kütüphane sistemi : Kitap, üyeler, türler, ödünç hareketleri
E-ticaret sistemi : Ürünler, müşteriler, siparişler, teslimat, fatura bilgileri, üreticiler, tedarikçiler, dağıtıcılar...
Futbol Ligi : Takımlar, sahalar, oyuncular, fikstür, hakemler, antrenörler
Okul Sistemi : Öğrenciler, öğretmenler, dersler, derslikler
Personel Sistemi : Çalışanlar, meslekler, çalışılan birimler, maaşlar, izinler
Sözlük : kelimeler, anlamlar, diller

Not : Tablolara isim verilirken mümkünse tekil isimler kullanılmalıdır. Böyle yapılırsa; hem daha anlaşılır bir tasarım yapılmış olur hem de daha sonra kodlama aşamasında karışıklığın önüne geçilmiş olur. Örneğin içinde Kitap ile ilgili bilgiler bulunduran tablonun adını Kitap koymak oldukça mantıklıdır.

2. Her nesne için bir tablo oluşturulur: Her nesne için bir tablo oluşturulur ve her bir tabloya içereceği veriyi en iyi anlatan bir isim verilir. Tablo oluşturma işi, bir kağıt üstünde sembolik olarak gösterilebilir veya doğrudan MS Access, SQL Server, MySQL, Oracle ... gibi kullanılmakta olunan VTYS üstünden de oluşturulabilir. Tüm proje bitirilinceye kadar bu tablolar üzerinde muhtemel değişiklikler yapılabilir.

3. Her bir tablo için bir anahtar alan seçilir: Veritabanındaki herhangi bir veriye erişilmeden önce tabloya erişilir. Bir veritabanında üzerinde en çok işlem yapılan nesne grubu genellikle tablolardır. Bu aşamaya kadar hangi tabloların oluşturulacağına karar verildi. Her bir tablonun içinde hangi bilgilerin saklanılacağı kabaca tasarlanır. Bu aşamada, tabloda yer alacak her bir kaydı bir diğerinden ayırabilecek bir sütuna ihtiyaç duyulur.

Örneğin bir kitap seçilmek istenildiğinde, bu kitabın hangi kitap olacağı öyle bir anlatılabilmeli ki, başka hiçbir kitap ile karışmamalıdır. Bunu yapmanın tek yolu, bir alanı birincil anahtar alan olarak belirlemektir. Anahtar alan seçilirken, kısıtlamadığı sürece, doğal alanlar seçilmeye dikkat edilmelidir. Örneğin araçlar ile ilgili bir tablo yapılırken, plakalar anahtar alan olarak belirlenebilir. Çünkü her bir plakadan bir tek araç trafiğe çıkabilir ve plakalar kısıtlamaz. Öğrenci tablosu için, öğrenci numarası doğal bir anahtar alandır çünkü aynı okulda, aynı numaradan bir öğrencinin daha bulunması söz konusu değildir. Personel tablosu için, personel sicil numarası doğal bir anahtar alandır çünkü aynı işyerinde, aynı numaradan bir personel daha bulunmaz.

Kitap tablosu için ISBN numarası anahtar alan olarak tanımlanabilir ama, aynı kitaptan iki adet olduğunda, ISBN numarası bizi kısıtlar. Elimizde iki adet "Önümüzdeki Yol" kitabı varsa, her iki kitabın da ISBN numarası aynıdır. Kitaplardan birisi eski diğeri yeni olabilir. Bu bir kargaşaya neden olabilir. Çünkü eski kitabı kime, yeni kitabı kime verdiğimizin takibini ISBN numarası ile yapmak mümkün değildir. Ancak bir E-Ticaret sitesi tasarlanırken, stoktaki tüm kitaplar birbiri ile eşdeğer olacağından ya da öyle olduğu varsayıldığından ISBN numarası birincil anahtar alan olabilir. Bu durumda, adet diye bir niteliğin aynı tabloda yer alması gerekecektir.

4. Nesnelerin gerekli her bir özelliği için tabloya bir sütun eklenir: Tablo adları tanımlandıktan ve anahtar adları belirlendikten sonra, tablolara sırasıyla adını veren nesnelerin her bir özelliği için bir alan (sütun) eklenir.

Örneğin, **kitap için;** Kitap no, ISBN no, kitap adı, yazarı, türü, sayfa sayısı, özeti, fiyatı, baskı yılı...

Üye için; UyeNo, adı, soyadı, e-mail adresi, ev telefonu, cep telefonu, iş telefonu....

Personel için; Personel sicil No, adı, soyadı, e-mail adresi, mesleği, çalıştığı birim, maaş....

Bu hazırlıklar yapılırken yapılması istenilen proje ile ilgili basılı formlar vs. varsa, onların incelenmesi tabloya eklenecek sütunların hangi özellikler olması gerektiği konusunda karar verilmesinde yardımcı olurlar.

İPUCU : 1. En başa birincil anahtar olarak belirlenen alanı eklemek bir kural değildir, ancak tablonun anlaşılırlığı ve göze hoş görünmesi açısından tercih edilmesi faydalı olacak bir tekniktir.

2. Genellikle, yapay birincil anahtar alanlar tablo adı ile başlar ve sonunda ID vardır. Ogrenci tablosu için ogrenciID, Personel tablosu için personelID gibi.

5. Tekrarlayan nesne özellikleri için ek tablolar oluşturulur : Akılda hep şu soru olmalıdır: veri tekrarı olacak mı? Veri tekrarı olacaksa bir yerlerde hata yapılıyor demektir. Bu durumda eldeki tablonun en az bir tabloya daha ayrılması gerekiyor demektir.

Şu da unutulmamalıdır, her projeye uyacak evrensel bir veritabanı tasarım tekniği yoktur. Yani her şey belli kurallar çerçevesinde ne kadar detayıyla düşünülüp tasarlandığına bağlıdır.

Örneğin, her bir kitap için tür belirledik ama, bir kitap hem kişisel gelişim kategorisine hem de hikaye kategorisine girebilir. Ya da e-ticaret sisteminde bir ürünün birden fazla reyonda yer alması gerekli olabilir. Veya bir kitap birden fazla kişi tarafından yazılmış olabilir. Bir kitap için birden fazla türü kaydedebilme ele alınsın:

Bu türden bir sorunu çözmek için ilk akla gelen şey, Kitap tablosunda tür alanı için 2.sütun daha eklemek olabilir. Bu tabloya 2.Tür ve 3.Tür diye iki sütun alanı daha eklemek. Ama çoğu kitap bir tek türdendir ve bu kitap için eklenen 2 alan hep boş kalacaktır. Öte yandan, 4.türe birden giren bir kitap olduğunda 4.tür bilgisi nereye yazılacaktır? Aynı alana mı? Ya da dört adet bölüm mü açılacak? Bunlar, veritabanı tasarımının doğasına terstir.

2.Çözüm yolu ise, bir kitabı iki kere kaydedip, birincisini, 'Kişisel Gelişim' türü olarak; ikincisini de 'Hikaye' olarak girmektir. Bu durumda tabloda aynı kitaba ait iki kayıt olacaktır ve kitap türü dışındaki diğer tüm bilgiler tekrar edecektir. Ya da bir süre sonra, kitap hakkında girilen bilgilerin yanlış olduğu fark edildi. Hangi kayıt güncellenecektir? Ya biri düzeltip diğeri unutulursa? Sonuçta veri tekrarı ve veri bütünlüğünün bozulması söz konusudur.

Bu da yine ilişkisel veritabanı tasarımının doğasına terstir. Bu durumda, türler diye bir yeni tablo oluşturup, bir de kitap_turler diye 2.tablo ' yu oluşturduktan sonra bu türden bilgileri burada tutmak gerekecektir. Böylelikle, hiçbir türde yer almayan kitaptan 10 ayrı türde yer alan kitaba kadar bütün olasılıklar için bir çözüm geliştirilmiş olur.

Aynı işlem öğrenci ve öğrenciye ait ders notları için düşünülebilir. Öğrenciye ait ders not bilgilerinin yazıldığı tabloya ait sütunların aşağıdaki gibi olduğunu varsayılırsa;

ÖğrenciNo	DersinAdı	VizeNotu	FinalNotu	Ortalama

Bir öğrenci aldığı dersten başarılı olursa vize ve final notu yazılarak ortalaması hesaplanır ve sorun yaşanmaz. Ama öğrenci bu dersten başarısız olursa bu dersi yeniden almak zorundadır. Yeniden aldığı bu derse ait ders notlarının nereye yazılacağının düşünülmesi gerekir. Eski notlarının da kalması gerektiği düşündüğü bu durumda tablo aşağıdaki gibi tasarlanabilir.

ÖğrenciNo	DersinAdı	VizeNotu	FinalNotu	Ortalama	VizeNotu	FinalNotu	Ortalama
03101001	BILGISAYAR	37	40		45	48	

Tabloda 2 adet not yazılabilecek alan vardır. Peki ama öğrencinin dersi ikiden fazla kere tekrar etmesi gerekirse ne olacak? Bu durumda yeni sütun alanları mı eklemek gerekecek? Tabloya 3 tane not yazma alanı eklendiğinde dersi bir kere alan ve başarılı olan öğrenciler için 2. ve 3.alanlar boş kalacaktır. Bu her öğrenci için değişebilecek bir durum olduğu için tablo tasarımında bu mantıkla düşünmek doğru değildir. Yukarıda ki örnekte de açıklandığı gibi bu şekilde bir tasarım yapılmaz.

Ayrıca; tabloda tanımlanan her sütun alanı, bu alana hiçbir bilgi yazılmasa bile HD'de yer kaplayacağı için;diskte tanımlanan bu alanlar boşuna kullanılmış olacaktır.

Dolayısı ile diskte de boş yere alan işgal edilmiş olacağından tabloda gereksiz sütun alanlarının tanımlanmaması gerekir. Örneğin, tabloda gereksiz tanımlanan bir sütun alanı diskte 4byte yer kaplıyor ise ve tabloda toplam 15 bin öğrenci var ise; gereksiz kullanılan toplam HD alanı 4 * 15.000 = 60.000 byte olacaktır. Sadece tek bir alan için bu kadar alanın boş yere kullanılmış olması hoş bir durum değildir.

Bu durumda tablo tasarımında yapılması gereken düzenleme aşağıdaki gibi olmalıdır. Bir öğrenciye ait dersler yazılırken alt alta satırlar şeklinde kayıt (record) olarak yazılarak yapılmalıdır.

ÖğrenciNo	DersinAdı	VizeNotu	FinalNotu	Ortalama
03101001	BILGISAYAR	37	40	
03101002	INGILIZCE	56	58	
03101001	BILGISAYAR	45	48	
03101001	BILGISAYAR	69	78	

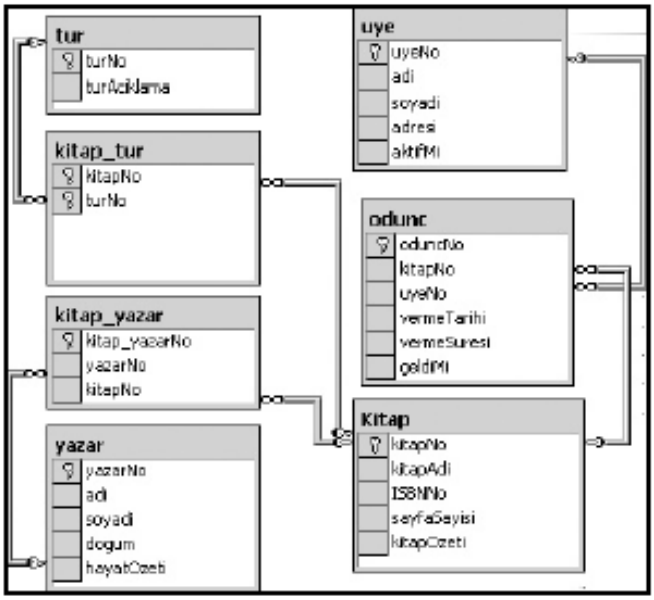
Doğru tablo tasarımı ve kayıt girişi yukarıdaki tabloda olduğu gibi olmalıdır. Burada 03101001 numaralı öğrencinin BILGISAYAR dersine ait notları bu tablodan ilerleyen bölümde anlatılan SQL cümleciği ile seçilerek bulunabilir.

6. Anahtar Alana Bağlı Olmayan Alanlar Belirlenir : İlişkisel veritabanında, tablodan herhangi bir tek kayda erişmek için mutlaka bir farklı özellik sağlanmalıdır ve bu özellik de anahtar alan tarafından sağlanır. Ancak bazen, anahtar alan ile aynı satırda yer aldığı halde, anahtar alan ile birebir ilişkisi olmayan bir alan yer alabilir. Bu türden alanların elimine edilip ayrı tablolara ayrılması gerekir. Örneğin, ödünç tablosu ele alınacak olursa, ödünç verilen her kitap için ödünç alanın adresi de bilinmek istenirse, bu ödünç tablosuna yazılamaz. Çünkü ödünç tablosunun birincil anahtar alanı oduncNo 'dur ve bu alan, ödünç verme işlemi ile ilgilidir. Oysa ödünç alanın adresi, ödünç alan kişinin kendisine bağlı bir özelliktir. Bu kişinin her aldığı kitap için adresini tekrar yazmaya gerek yoktur. Aynı şekilde otomasyon içerisinde başka yerlerde de bu kişinin adres bilgilerine muhtemelen ihtiyaç duyulabilir çünkü adres, üyenin bir özelliğidir.

Ödünç verilen kitabın adresi öğrenilmek istenildiğinde, üyeler adında bir tablo daha açılıp, burada herkesin adres bilgisi tutulmak zorunda kalınır. Ödünç tablosunun ise, oduncAlan bilgisi olarak, Üyeler tablosunun birincil anahtar alanına bir bağlantı (yabancı anahtar) içermesi daha doğru olur.

7.Tablolar arasındaki ilişkiler tanımlanır : Her biri bir nesneye dair özellikleri barındıran tabloların tümü göz önüne alınır ve birbirleri ile olan ilişkileri tanımlanmaya çalışılır. Örneğin kitabı ödünç verebiliriz. Bu durumda, ödünç tablosu ile Kitap tablosu ilişkili olacaktır. Kitap üyelere ödünç verilir. Bu durumda, ödünç ile üyeler arasında da bir ilişki vardır. Türler ile Kitap arasında bir ilişki vardır, bir kitabın en az bir türe dahil olması gerekir.

Bu projedeki nesneler (tablolar) arasında ilişkiler aşağıda yer almaktadır:



Şekil 4.1.1. Örnek Projenin SQL Server 2000 deki diyagramı

İlişkili her iki tablo bir birincil alan ve bir yabancı anahtar alan üstünden birbirine bağlanır. Aynı diyagramın bir benzeri Ms Access veya Oracle veri tabanlarında da hazırlanabilir.

Farklı tablolardaki iki alan aynı veriyi tutuyorsa, iki alana da aynı adı vermek, karışıklığa yol açabilir gibi görünse de aslında daha düzgün bir yapı ortaya çıkar. KitapNo alanı kitap tablosunda da ödünç tablosunda da kitap numarasını tutmaktadır. Bu alanlardan birine KitapNo, diğerine oduncGidenKitapNo demek, kafa karışıklığına neden olabilir. En önemlisi de her alan için her tabloda farklı isimler kullanmak değişkenlerin isminin akılda tutulmasını zorlaştıracak ve daha sonraki tablolar üzerinde işlem yaparken işlemleri zorlaştıracaktır. Her seferinde ilgili alanın hangi isimle kaydedildiğine bir listeden bakmak zorunda kalınacaktır. Çünkü büyük bir veritabanı projesinde 250 den fazla tablo bulunabilir. Her tabloda da bir çok alanın bulunacağı dikkate alındığında her alana ait isimlerin akılda tutulması mümkün olmamaktadır. Birden fazla tabloda olan alanlar için; aynı ismi kullanmak bu zorluğu ortadan kaldıracaktır. En mantıklısı her ikisine de KitapNo demektir.

4.2. Veri Tabanı Normalizasyonu

Aslında ilişkisel veri tabanı tasarımından ziyade, bir tablo içerisinde yer alacak kaydın nelerden oluşmasına karar vermeye yarayan normalizasyon kuralları başlı başına bir işlemdir. Normalizasyon; veritabanı tasarım aşamasında gerekli bir işlem olduğundan bu bölümde incelenecektir. Genel kabul görmüş 5 normalizasyon kuralı vardır. Burada her bir kuralı tam olarak anlatmak mümkün değildir. Ancak bu kurallar, ilişkisel veritabanının tanımı ile birlikte ortaya konulmuştur. Özet olarak fikri vermesi açısından normalizasyon kurallarına aşağıda yer verilmiştir.

1. Normalizasyon Kuralı :

Bir satırdaki bir alan yalnızca bir tek bilgi içerebilir. Birden fazla yazarı olan kitap için yazar1, yazar2 ve yazar3 diye alanların açılması ile bu kurala uyulmamış olunur. Böyle bir durumda, ayrıca yazarlar tablosu da oluşturularak kural çiğnenmemiş olur.

Veri tabanı tasarımında; verileri virgül veya bir başka karakter ile ayrılıp aynı alana girilmesi ve daha sonra program içerisinde split ile bu değerlerin ayrılması genellikle sık yapılan hatalardan birisidir. Ancak bu ilişkisel veritabanının doğasına terstir. Bunun yapılmaması gerekir.

2. Normalizasyon Kuralı:

Bir tablo için, anahtar olmayan her alan, birincil anahtar olarak tanımlı tüm alanlara bağlı olmak zorundadır. Örneğin, Ödünç tablosuna KitapAdi diye bir alan eklense idi, bu sadece ödünç verilen kitap ile ilgili bir bilgi olacaktı ve oduncNo 'na bağlı bir nitelik olmayacaktı. Bunu çözmek için, kitap adları ayrı bir tabloda tutularak sorun çözülebilir.

Ya da anahtar alanın birden fazla alandan oluştuğu tablolarda, anahtar alanlardan sadece birine bağlı veriler tabloda yer almamalı, ayrı bir tabloya taşınmalıdır. Bunun tersi de geçerlidir. Yani iki ya da daha fazla tablonun birincil anahtarı aynı olamaz. Böyle bir durum söz konusu ise, bu iki tablo tek tabloya indirilmelidir.

3. Normalizasyon Kuralı:

Bir tablo için, anahtarı olmayan bir alan, anahtarı olmayan başka hiç bir alana bağlı olamaz. Örneğin, kitaplar için cilt tipi adında bir alan eklenip burada da karton kapak için K, deri cilt için D, spiral cilt için S yazılsaydı, bu kodlama, kitap tablosunun birincil anahtarı olan kitapNo alanına bağlı bir kodlama olamazdı. Çünkü bu kodlama bir başka anahtarı olmayan alana bağlıdır. Bunun sonucunda da veritabanında, karşılığı olmayan bir kodlama yer almış olurdu. Cilt tipi bilgisini kodlu olarak tutan alan aslında cilt tipi açıklaması olan başka bir alana bağlıdır. Bu ilişki başka bir tabloda tutulmalıdır. Bu durumda, cilt şekillerini tutan bir tablo açılması gerekir. Bu tablonun alanları da ciltTipKodu ve ciltSekli olabilir. Ancak bundan sonra, kitaplar tablosunda ciltTipi adında bir sütun açıp buraya da D,S,K gibi kodlar yazılabilir.

4. Normalizasyon Kuralı:

Birincil anahtar alanlar ile anahtarı olmayan alanlar arasında, birden fazla bağımsız bire-çok ilişkisine izin verilmez. Örneğin, tabloda yer alan bir kitap, hem hikaye kitabı hem de kişisel gelişim kitabı olabilir. (Bu durumda kitabın adı, kişisel gelişim hikayeleri olurdu her halde) Bu durum Kitap tablosunda nasıl ifade edilebilir?

4.Normal formu sağlamak için, her bağımsız bire çok ilişki için ayrı bir tablo oluşturulması gerekir. Bu örnekte, türler için yeni bir tablo açılması gerekir. Tablonun adına türler denilebilir. Daha sonra kitapTurleri diye bir başka tablo daha açılması gerekir. 'Kişisel Gelişim Hikayeleri' adlı kitap için, öncelikle kitap numarası, Hikaye bölümünün kodunun yer aldığı bir satır; ardından da yine kitap numarası, ardından da kişisel gelişim türünün kodunun aldığı yeni bir satırın daha eklenmesi gerekir.

5. Normalizasyon Kuralı:

Tekrarlamaları ortadan kaldırmak için her bir tablonun mümkün olduğunca küçük parçalara bölünmesi gerekir. Aslında ilk 4 kural sonuçta bu işe yarar ancak, bu kurallar kapsamında olmayan tekrarlamalar da 5 normalizasyon kuralı ile giderilebilir.

Örneğin, kitaplar için bir edinme şekli bilgisi girilecek sütun eklenmek istenebilir: Bu bölüme girilebilecek bilgiler bellidir: Bağış veya satın alma.

Bu bilgiler başka bir tabloda tutulabilir. Böylelikle, kullanıcıların bu alana gelişi güzel bilgiler girmesi engellenmiş olur. Bu da sorgulama esnasında veriler arasında bir tutarlılık sağlar. Bu işlem sonucunda, tutarsızlıklara neden olabilecek ve sık tekrarlayan veriler başka bir tabloya taşınmış olur. Bu tablo için, veritabanı programlamada 'look-up table ' terimi kullanılır.

Ancak, veritabanı normalizasyon kuralları, bir ilişkisel veritabanının tasarlanma aşamalarını değil de ilişkisel veritabanında yer alacak kayıtların ilişkisel veritabanı ile uyumlu olup olmadığını denetlemeye yöneliktir. Özetle ilişkisel bir veritabanı tasarımı şu dört öğeyi barındırmalıdır.

1. Veri tekrarı yapılmamalıdır.
2. Boş yer mümkün olduğunca az olmalıdır.
3. Veri bütünlüğü sağlanmalıdır.
4. Veriler, aralarında bir ilişki tanımlanmaya müsait olmalıdır.

4.3. İlişkisel Veri Tabanı Yönetim Sistemleri

Veritabanı Yönetim sistemlerinden günümüzde kullanımı en yaygın olan ilişkisel veritabanıdır ve en yaygın veritabanı yönetim sistemleri, ilişkisel Veritabanı Yönetim Sistemleri ' (VTYS) dir. İlişkisel veritabanının en önemli yanı, tablolardan oluşmasıdır. Daha önemli yanı da bu tabloların birbiri ile ilişkilerinin olmasıdır. VTYS 'lere "ilişkisel" denmesinin anlamı budur.

Bir veritabanında ilişkiden söz edebilmek için en az iki tablonun yer alması gerekir ve bu iki tablodaki verilerin birbiri ile bir şekilde ilişkilendiriliyor olması gerekir. Yine bir önceki örnek olaya dönecek olursak, Kitap listesi ile ödünçler listesi arasında bir ilişki vardır. Çünkü Kitap listesinde olmayan bir kitap bizde yoktur ve ödünç verilemez. Haliyle de mantık olarak bu türden bir ödünç bilgisi ödünç listesinde yer almamalıdır. Olaya tersten bakılacak olursa, geri dönmeyen bir kitap hakkındaki detaylar öğrenilmek istenildiğinde ödünç listesindeki kitap numarası alınır. Daha sonra aynı numaraya karşılık gelen kitap, Kitap tablosundaki satırda bulunur. Bu satırdaki bilgiler, bize kitap hakkındaki tüm detayları verir.

Kitap tablosundaki kitapNo alanı aday anahtar (indeks)'tir. Odunc tablosundaki KitapNo alanı, 'yabancı anahtar ' (foreign key) alandır, çünkü Kitap tablosundaki bir kaydı sembolize etmektedir. Tüm bunların ardından VTYS 'leri hakkında özet olarak diyebiliriz ki;

Bir Ilişkisel Veritabanı Yönetim Sistemi tablolar üstünde şu üç işlevi yerine getirmek zorundadır.

1.Seçme : Herhangi bir tabloda (listede) yer alan tüm bilgileri gösterebilmelidir. Örneğin, Kitap tablosunun bir dökümünü verebilmelidir ya da kitap listesinden bazı kitapların bilgilerini getirip diğer bir kısmını getirmeyebilmelidir.

2. İzdüşürme : Herhangi bir tablodan sadece belli sütunların yer aldığı seçme işlevlerini yerine getirebilmelidir. Örneğin, canı isteyen bir kullanıcı kitabın sadece adını ve kaç sayfa olduğunu seçebilmelidir.

3.Birleştirme : Birden fazla tabloda yer alan bilgiler, yeri geldiğinde tek bir tabloymuş gibi sunulabilmelidir. Örneğin, ödünç alınıp da geri getirilmeyen kitapların adları ve kimler tarafından alındığı bir tek tabloymuş gibi gösterilebilmelidir.

VTYS bu 3 temel işlevi yerine getirebilmelidir. Bunlardan üçü, ikisi veya biri aynı anda yerine getirilmek durumunda kalınabilir. Örneğin, sayfa sayısı 200 'den büyük kitapların sadece ismi görülmek istenirse, hem izdüşürme hem de seçme işlemine ihtiyaç duyulur. Veriler ve depolanma şekilleri farklı olabilir. Önemli olan, VTYS'nin SQL ile yönetilebilir olmasıdır. Böylelikle, verilerin bilgisayarda fiziksel olarak ne şekilde depolandığı, kullanıcı bilmek zorunda değildir.

Yani, kullanıcı temel veri saklama işlem ve yöntemlerinden izole edilmiş olur. Kullanıcının verileri etkili olarak kullanması için bilmesi gereken tek şey SQL olmalıdır. SQL ilerleyen bölümlerde anlatılmıştır.